SGTC FORM

stryker

TITLE:	**Software Architecture Design**
DOCUMENT NUMBER:	**SGTC-QFM-DLC-001-04** **VERSION:** **07**

Software Architectural Design Approval:

Issued by: DEV	Deepak Sharma	Deepak Sharma *Electronically signed by: Deepak Sharma Reason: I am signing as the Author of this document Date: Mar 31, 2022 09:48 GMT+5.5*	31-Mar-2022	
	Name printed	Signature	Date	
Revision No.:				
Approved by PL:	Akhil Gupta	Akhil Gupta *Electronically signed by: Akhil Gupta Reason: I approve this document Date: Apr 1, 2022 09:15 GMT+5.5*	01-Apr-2022	
	Name printed	Signature	Date	
Approved by TEST:	Pragya Nidhi	Pragya Nidhi *Electronically signed by: Pragya Nidhi Reason: I approve this document Date: Apr 1, 2022 11:30 GMT+5.5*	01-Apr-2022	
	Name printed	Signature	Date	
Approved by QA:	Sreejith Viswam	Sreejith Viswam *Electronically signed by: Sreejith Viswam Reason: I approve this document Date: Apr 2, 2022 09:33 GMT+5.5*	02-Apr-2022	
	Name printed	Signature	Date	
Approved by IT:	NA			
	Name printed	Signature	Date	
Approved by OE	Vikram Puri	Vikram Puri *Electronically signed by: Vikram Puri Reason: I approve this document Date: Apr 1, 2022 13:57 GMT+5.5*	01-Apr-2022	
	Name printed	Signature	Date	

Project Number: SGTC-NPD-001, Project Name: SmartMedic Phase II

System Name: SmartMedic

Version Number: 00

Document ID: D001020031

SOFTWARE ARCHITECTURE DESIGN

Nurse Station (Cloud and Web)

Table of Contents

1. INTRODUCTION

1.1 Purpose

This document describes the overall architectural design of the **Nurse Station Application and Stryker's Cloud Infrastructure.** It depicts from a high-level perspective the system's context and scalable structure. Where details go beyond the scope of this document the reader is referred to lower-level design. The intended audience for this document is the development team and the quality assurance team.

1.2 Scope

The Nurse Station Application is intended to be used as an accessory for medical beds in hospitals to measure patients' weight and alert the caregivers about patients' turn. The SmartMedic device attached to the bed reports all the data to the system, which gets displayed on the Nurse Station App. The application also displays graphs for patient's weight and turn data for better visualization of trends in data. The Nurse Station Application along with SmartMedic devices is intended to be used in an Indian hospital environment.

The architecture design for the following is included in this document:

- Stryker's Cloud Infrastructure
- The Nurse Station Application

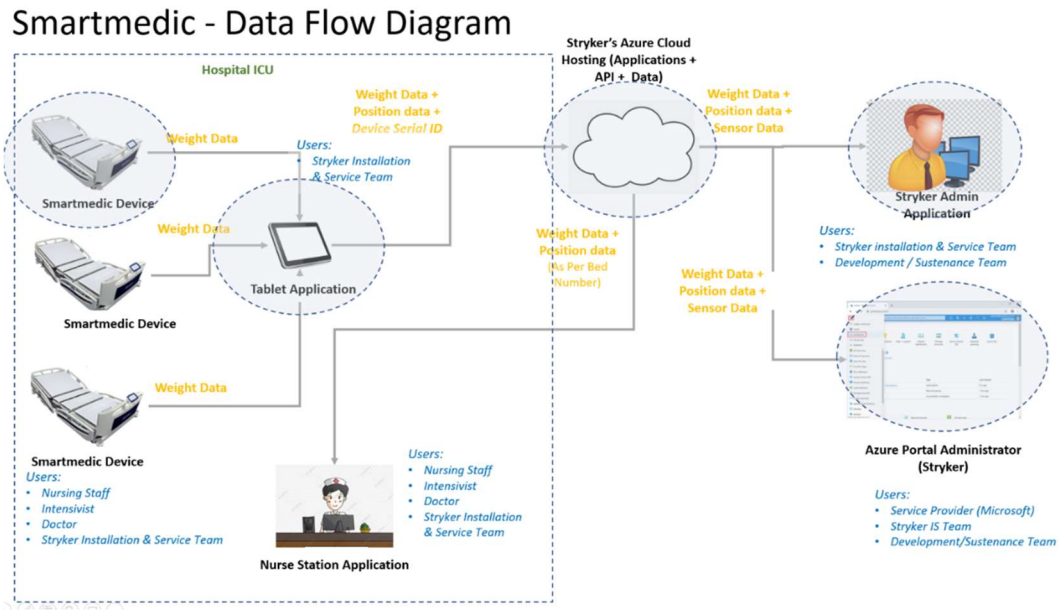

Figure 1: System Overview

1.3 Business Context

Following is the document # for reference:

Name	Reference number
Project Charter	D00001 Rev 01

1.4 Definitions, Acronyms, and Abbreviations

Following definitions, acronyms, and abbreviations are used throughout this document:

Term	Definition
IoT	Internet of Things
GUI	Graphical User Interface
TLS	Transport Layer Security
UI	User Interface
SDK	Software Development Kit
UX	User Experience
NSA	Nurse station Application
API	Application programming interface
DPS	Device Provisioning Service
AES	Advanced Encryption Standard
JSON	JavaScript Object-Notation
RDBMS	Relational Database Management System
MQTT	Message Queuing Telemetry Transport

1.5 References

Document	Doc. Number	Description
SRS	D001020023	Software Requirement Specification for Nurse Station
URS	D001020022	User Requirement Specification

1.6 System Context

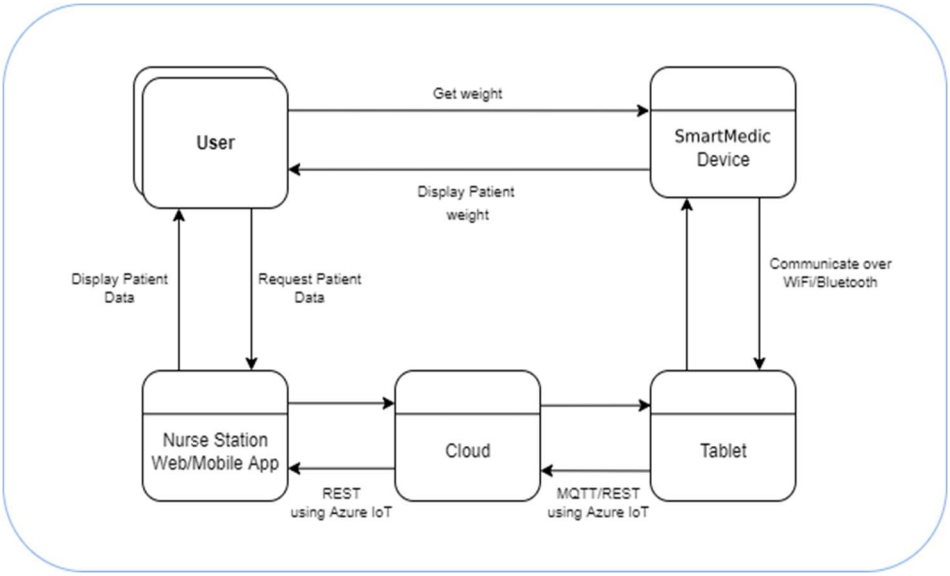

Figure 2: System Context View

The Healthcare professionals can request data from SmartMedic devices or can view patient history information on the Nurse Station Application after entering valid credentials. The Gateway Device Tablet Application is connected with a bedside device through Bluetooth/Wi-Fi, enabling the tablet to send real-time patient weight and position data to the Nurse Station Application. The data from the Gateway Device Tablet Application is continuously sent to the cloud backend where the system will use Azure Function to route the data to the database based on certain conditions like non telemetry and telemetry data. The Nurse Station Application will be used to monitor the weight, position data, and trend of weight and position change on SmartMedic devices.safety

Sub Systems	Description
Nurse Station Application	Nurse Station application is based on the NodeJs based REST API and ReactJs frontend framework and interacts with the Cosmos DB cloud infrastructure.
Stryker's Cloud Infrastructure	Microsoft Azure provides IoT cloud infrastructure. The Azure Internet of Things (IoT) is a collection of Microsoft-managed cloud services that connect, monitor and control IoT assets. The system interacts with the IoT Hub and Device Provisioning Hub features of Azure IoT.

2. SYSTEM DECOMPOSITION

2.1 Hardware Decomposition

This is not applicable as the Nurse Station Application and Cloud Infrastructure will be hosted on the services provided by Microsoft Azure.

2.2 Software Decomposition

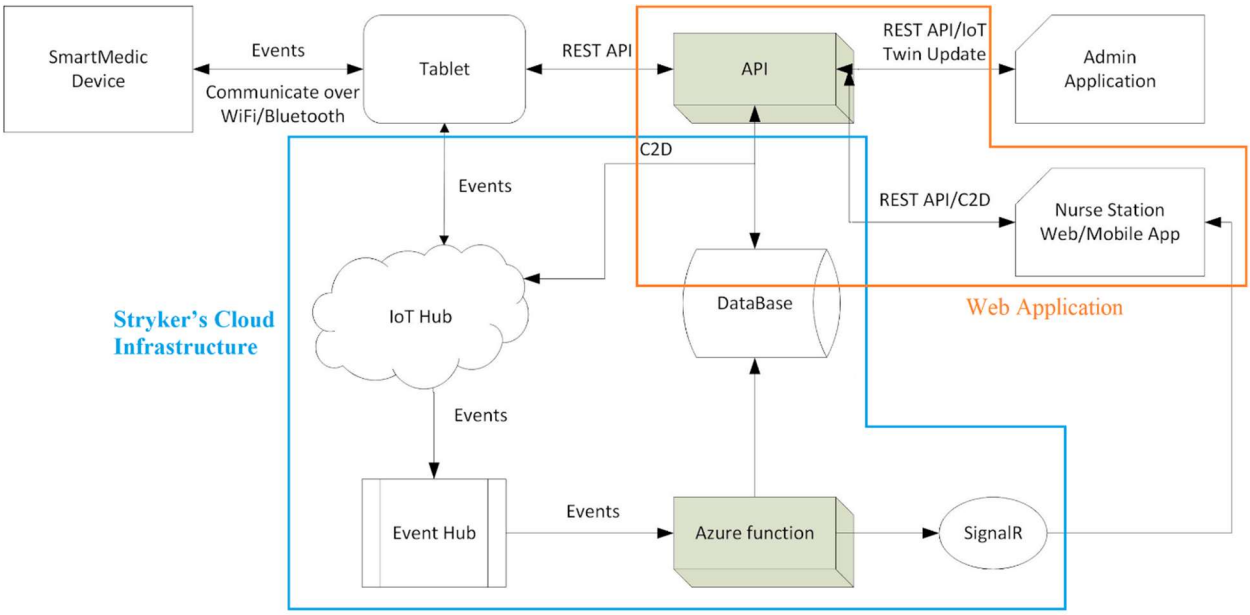

Figure 3: Software Detail View

Software Decomposition consists of the following components:

- Stryker's Cloud Infrastructure
- Web Application
  - Web Services
  - Web Interfaces

The sections below (2.2.1 and 2.2.2) will describe each component and its architecture individually.

For more details and protocol for communication from the Tablet to Cloud, please refer to Appendix A.

2.2.1 Stryker's Cloud Infrastructure

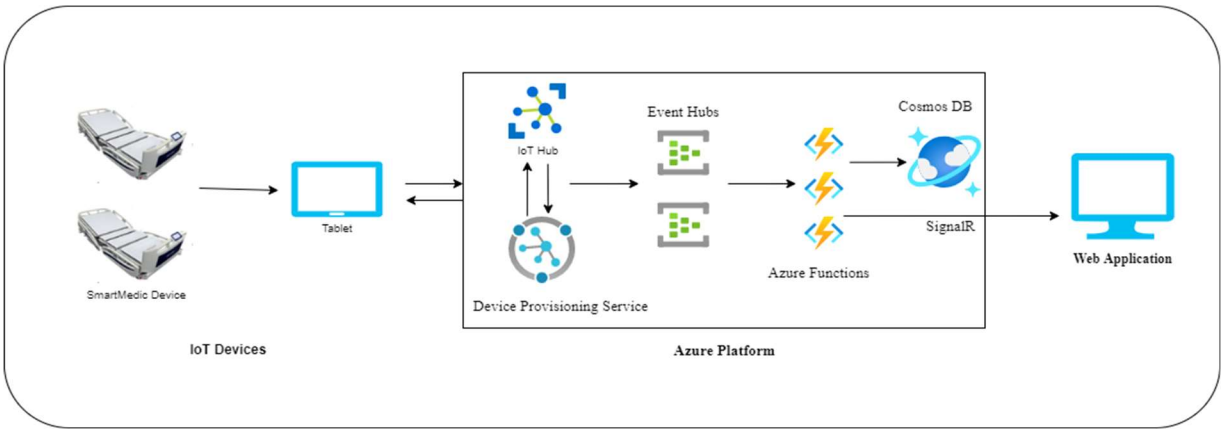

Figure 4: Cloud Infrastructure

Cloud infrastructure consists of the following components:

- IoT Devices
- Azure Platform
- Device Provisioning Service
- Event Hubs
- Azure Function
- SignalR
- Cosmos DB

2.2.1.1 IoT Devices

Function	Send events to the Tablet, which acts as a Gateway and forwards the data to the IoT Hub using MQTT protocol.
Input	Tablet parameters
Output	Events

2.2.1.2 Azure Platform

Function	Connection hub which shall receive data from the Gateway Device Tablet application and invoke corresponding triggers, configure the Tablet application to IoT Hub and enable telemetry transmission. The SmartMedic device sends data to the Tablet application which forwards it to the IoT Hub using MQTT protocol.
Input	Tablet Register Request, Device Telemetry (Tablet will send device info), Device Twin Updates (Tablet will send device info)
Output	Tablet Events, Tablet Telemetry Events, Tablet Connection String
Region	Southeast Asia

2.2.1.3 Device Provisioning Service

The IoT Hub Device Provisioning Service (DPS) is a helper service for IoT Hub that enables zero-touch, just-in-time provisioning to the right IoT Hub without requiring human intervention. DPS enables the provisioning of millions of devices in a secure and scalable manner.

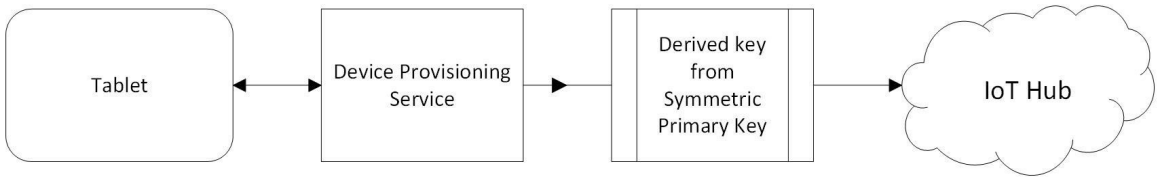

Figure 5: Device Provisioning Service (DPS)

2.2.1.4 Event Hub

Function	The Event Hub shall receive telemetry messages from the IoT Hub and invoke corresponding triggers.
Input	Telemetry Messages
Output	Device Telemetry Events
Configuration	Standard Tier, 20 Consumer groups, and 1000 Brokered connections
Region	Southeast Asia

2.2.1.5 **Azure Function**

Function	The Azure function shall receive device life cycle events such as DEVICE CONNECTED and DISCONNECTED from the IoT Hub and invoke corresponding triggers.
Input	Device Life Cycle Messages
Output	Device Life Cycle Events
Configuration	Event grid schema with webhook endpoint

The Azure function consists of the following major processors:

- Telemetry Processor
- Message Processor

Telemetry Processor

Function	Parse telemetry and store telemetry data. Default telemetry interval 5 seconds Update DB with telemetry data
Input	Device Telemetry
Output	DB record in cosmos DB
Region	Southeast Asia

Message Processor

Function	Parse event telemetry sent from the bedside device. • Admit event • Discharge event • Get Weight • Position • Position timer change • Usage Update DB with new weight and position Trigger turn signal with bed id using SignalR service.
Input	Event Telemetry
Output	DB record, SignalR signals
Region	Southeast Asia

2.2.1.6 SignalR

Function	Update server-side changes in clients that are connected to the server.
Input	Azure Functions
Output	Real-time signals for Gateway and web interface.
Configuration	Standard Tier, 1 Unit. Includes 1 unit with 100 concurrent connections per unit and 1,000,000 messages per unit/day
Region	Southeast Asia

2.2.1.7 Cosmos DB

It is a NoSQL Database used to store non-structured data in Azure as a high-performance, non-normalized database. It is flexible, with high availability and automatic scaling. It is used to store JSON documents.

Component	Database (Cosmos DB)
Function	Store telemetry in NoSQL format. Store application-specific data for client applications.
Input	Azure Functions, Web API
Output	DB Record
Region	Southeast Asia

For more details and protocol for communication from Tablet to IoT, please refer to Appendix B.

2.2.2 Web Application

2.2.2.1 Web Services (API)

Component	Web Services
Function	Provide web API interface for client applications (Web interface and Gateway App).
Input	Web Interfaces, Android App
Output	JSON response
Region	Southeast Asia

Note: Please refer to SRS for a list of web services and details of web services.

2.2.2.2 Web Interface (NSA)

Function	Provide web interface for monitoring beds.
Input	Read/Write URL
Output	Web Pages for user
Region	Southeast Asia

3. RISK CONTROL

As SmartMedic System is classified as Class A device, Nurse Station Application is also Class A the risk aspects are covered in the documents listed below.

Name	Reference number
Risk Table	D001020010
Risk Management Plan (RMP)	D001020007
Product Security Risk Table	D001020017
DFMECA (Nurse Station)	D001020096
DFMECA Tablet	D001020095
DFMECA Admin Application	D001020016

4. DEPLOYMENT VIEW

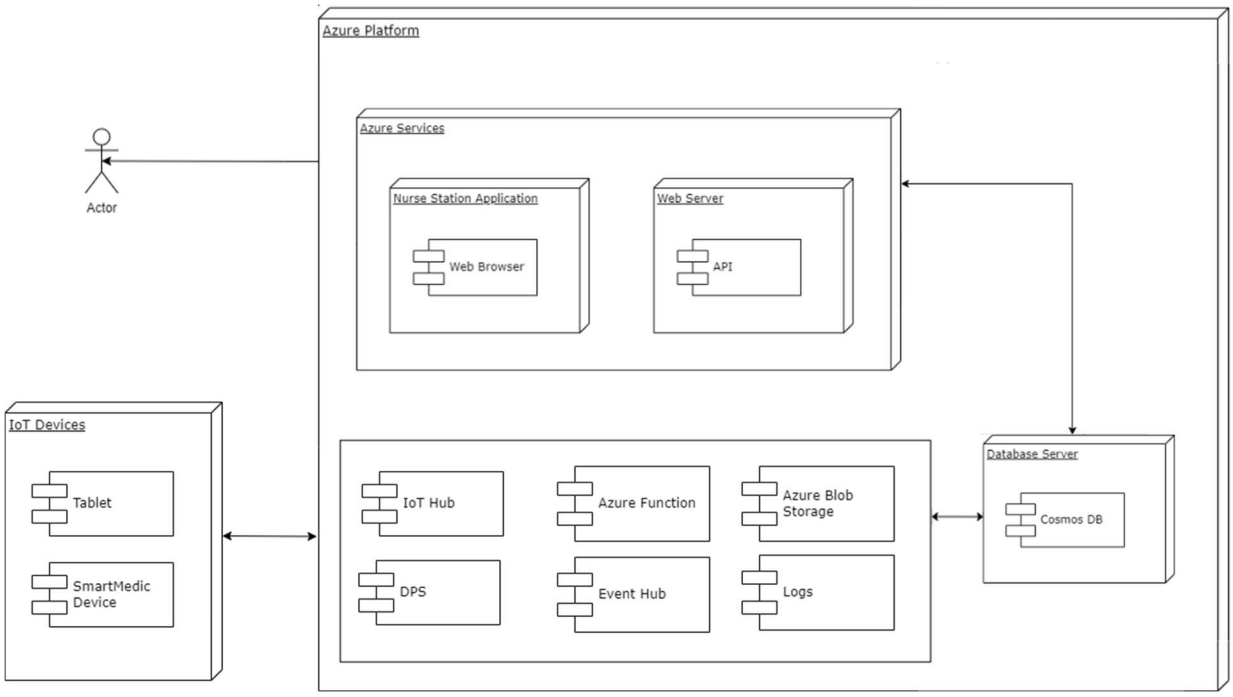

Figure 6: Deployment View

The Gateway Device Tablet Application is connected with a bedside device through Bluetooth/Wi-Fi which enables the tablet to act as a gateway for sending real-time patient weight and position data to the Nurse Station Application. The data from the Gateway Device Tablet Application is continuously sent to the cloud backend. The Nurse Station application will be used to monitor the weight, position data, dialysis data, and trends data obtained from SmartMedic devices in an ICU.

5. RUNTIME VIEW

The device, when initialized, sends a register event followed by an admit event which indicates that a patient has been admitted to that bed. Subsequently the device sends telemetry data every 5 seconds and a heartbeat event. The tablet acknowledges the event reception and forwards the events to the cloud. The device sends weight and tare events as configured and finally when the patient is discharged, it sends a discharge event.

5.1 Runtime View 1

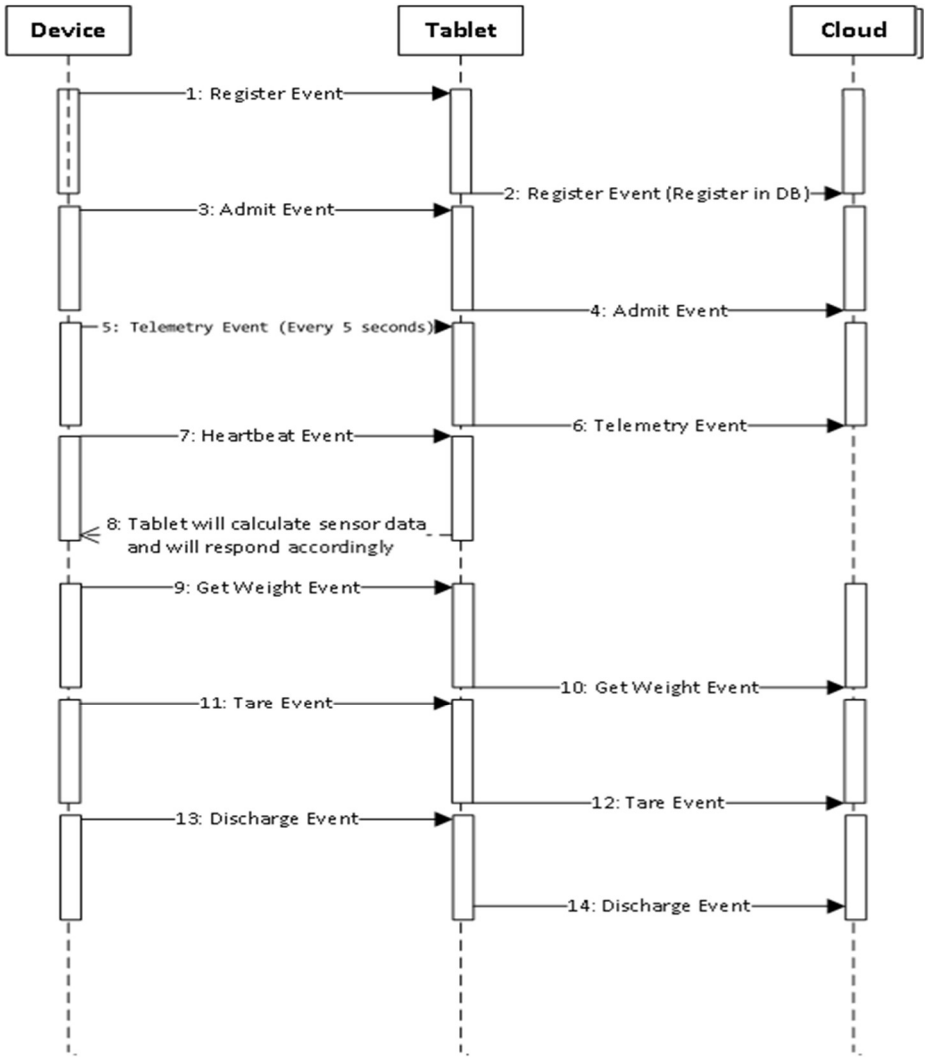

Figure 7: Device to Cloud Flow Overview

Devices will be sending the events to the tablet, and the tablet will pass the event data to the IoT Hub. Event Hub will fetch this data and pass it to the Azure function.

For device registration with the tablet, a new event (Device Register) will be required to create a mapping between Hospital, Tablet, and Device into the DB.

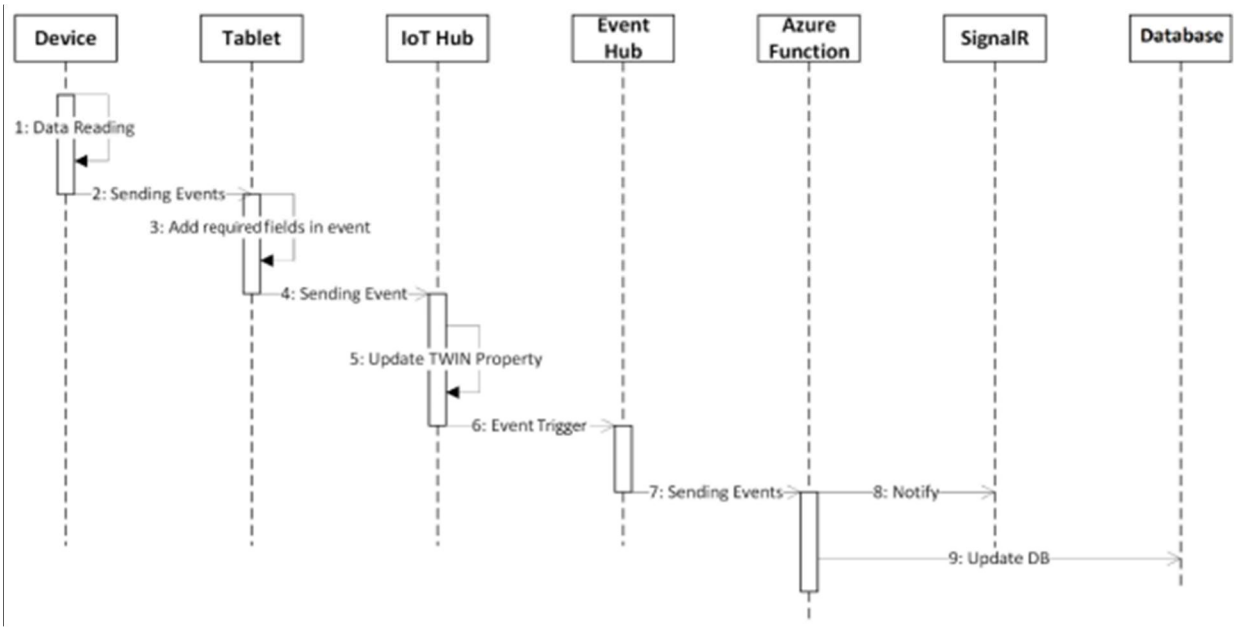

Figure 8: Event Flow

5.2 Runtime View 2

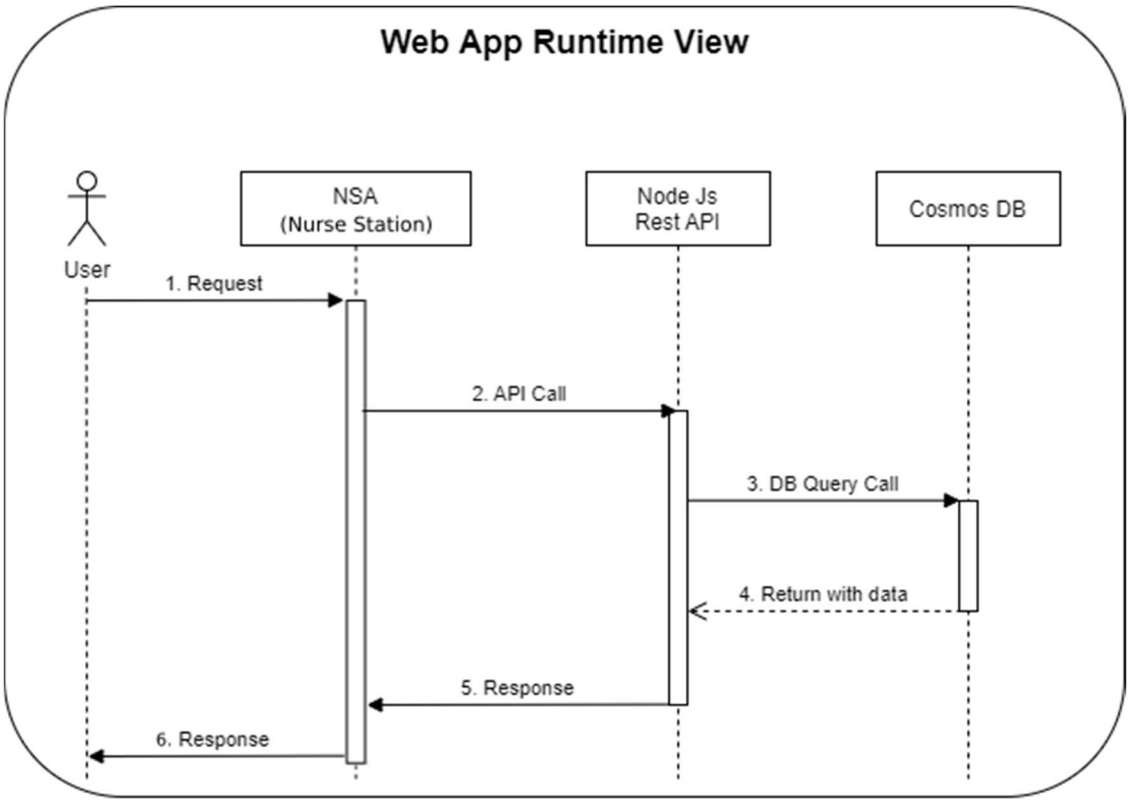

Figure 9: Web App Runtime View

5.2.1 Web Application (NSA)

The Nurse Station Application (NSA) is a web-based application developed in the ReactJs library. It will be used by authorized users to monitor ICUs, Beds, Patient's health-related data and get their reports accordingly. It requests data using REST API and will display the data in the form of cards and charts. It helps users to maintain the availability and patient data.

5.2.2 REST API

The REST API is developed using JavaScript base NodeJs and express library. It authenticates the client and accordingly fetches data from the Cosmos DB, and responds to the client with the pre-defined JSON format.

6. ARCHITECTURAL KEY ASPECTS

6.1 Safety

Nurse Station Application falls under software safety class A. The risks associated with Nurse Station are mentioned in the Risk File with document number D001020010.

6.2 Accuracy

The bedside application sends data to the frontend and this is just displayed in the User Interface. So, no modification or manipulation of data happens either at the frontend, backend or the database. This ensures that accuracy is maintained at the frontend.

6.3 Extensibility

The Nurse Station Web Application and the Nurse Station Web Services have component-based architecture and use various node modules that can be changed/modified if needed. With the benefit of modularity, any number of components can be added or removed from the application without impacting other non-dependent components. The Cloud infrastructure is also setup in such a way that the services can be extended or scaled when needed. For example, if need arises for larger database storage, that can be achieved by increasing the storage capacity.

6.4 Configurability

As the Nurse Station Web Application and Web Services have component-based architecture, the system is highly configurable. For example, if the authentication layer in Web Services needs to be replaced with a newer version, it can be achieved by modifying a few lines of code. Similarly, the Cloud infrastructure is also configurable for instance-level changes such as utilizing more CPU during high traffic and expanding the storage when needed.

6.5 Maintainability

The system is designed to be maintainable as the different components of the system can be deployed and maintained separately without impacting the other components. The deployment of the components is automated after changes are pushed to the repository.

6.6 Testability

The system is designed as per the standard component-based architecture which enables the individual components of the system to be tested separately and the entire system as well to can be tested together.

For more details on how the system is tested please refer to Software Verification Plan Document with document number D001020108.

6.7 Security

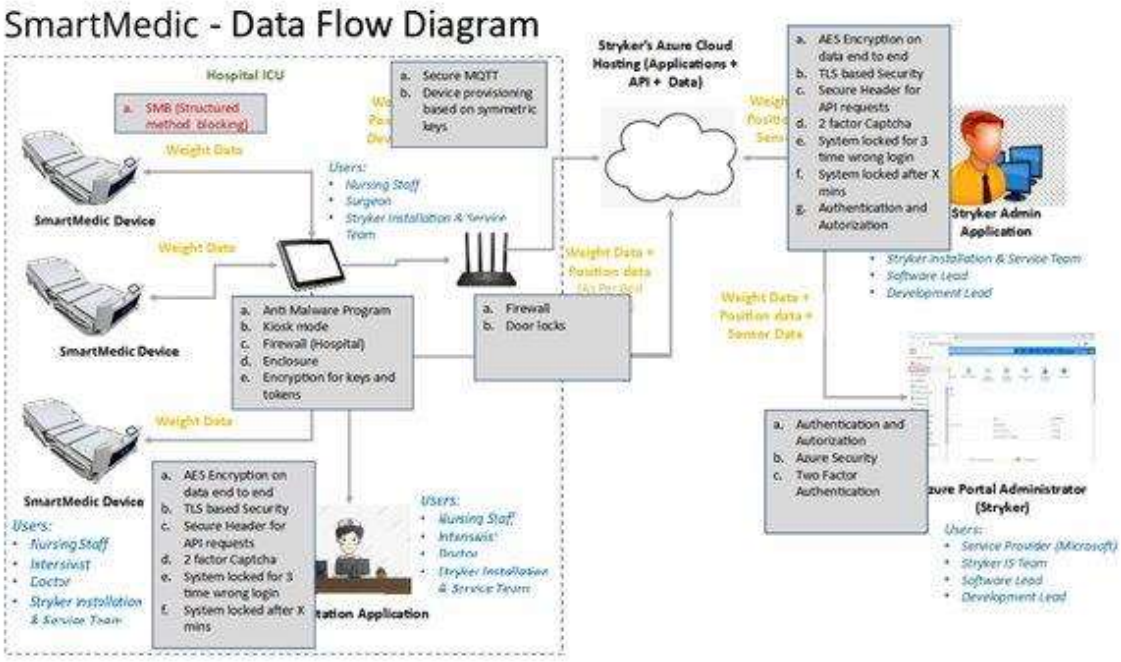

Figure 10: Product Security Diagram

The system uses end to end data encryption using AES 256-bit algorithm along with TLS based security and Secure headers for all the API requests. Moreover, 2-factor captcha authentication is also implemented while logging into the Nurse Station. The Nurse Station Web Service sits behind a firewall and door locks the IP-based attacks on the system.

- **User shall be able to view data only after authorization**

  When a user tries to access the website, it will ask for a Hospital Code which is unique for each hospital. Hence only authorized persons of the specific hospital can have access to this system.

- **Authorized Nurse Station shall present the data of the specific ICU to the hospital staff.**

  Once the authorization is done with the use of Hospital Code, only the person who has that Hospital Code can access a specific ICU. This is because in the settings, the ICU is assigned to the hospital. So once the authorized person logs in, he/she can have access to that ICU.

- **User shall be able to view the 'turn status' of each bed in a particular ICU on an authorized Nurse Station.**

  Once the authorization is done with the use of Hospital Code, only the person who has that Hospital Code can access a specific ICU and the beds in the ICU. Since the beds are assigned to an ICU which in turn is assigned to the hospital in the settings, once the authorized person logs in, he/she can have access to a particular ICU and its beds and can view the turn status of each bed.

- **User shall be allowed to view weight & position log of a bed on authorized Nurse Station of the ICU**

  Once the authorization is done with the use of Hospital Code, only the person who has that Hospital Code can access that specific ICU and beds. This means that they automatically can see the weight and position data, because in the Settings ICUs and beds are assigned to the hospital. So once the authorized person logs in, he/she can have access of the ICU, Beds and weight and position data as well.

- **User shall be allowed to raise a service request for any device placed in the ICU via an authorized Nurse Station**

  As the system is secure from top-level, if the user has login details for a hospital i.e., the Hospital Code, then he/she can have access to the ICU, Beds, weight, and position data. The user can also generate service requests if required.

- **Nurse Station shall display weight log of the entire duration of patient weight recording**

  As the system is secure from top-level, if you have login details for a hospital, i.e., the Hospital Code then you can have access to ICUs, Beds, weight, and position details. Such authorized users can also access the weight logs of patients.

6.8 Performance

The system is designed with Azure cloud. When the system has traffic, the Azure cloud will automatically adjust the CPUs and RAMs. When the system has multiple region traffic, Azure cloud automatically does horizontal scaling and whenever required the system also does vertical scaling. The system is designed with component-based architecture keeping performance in mind.

In the Nurse Station Application development, cutting-edge technology ReactJs is used. ReactJs comes with

- Virtual DOM
- Common components
- Unidirectional flow
- State management (Redux – as a Middleware)

In Web service (API) NodeJs is used, which is:

- Asynchronous and Event-Driven
- Very Fast
- Single-Threaded but Highly Scalable
- No Buffering

6.9 Scalability

Microsoft Azure Cloud Platform allows scaling of all the cloud components based on increase in load from devices or users. The system can automatically scale as per the load on the system.

As the load of devices increases the cloud infrastructure components scale automatically to handle load from devices.

In case of web application and web service, with the increase in number of parallel users using the web application the app service scales horizontally to cater to the increase in load.

6.10 Reliability

The patient's weight and position / turn data are calculated by algorithms that are tested. This data is stored in the database unchanged and the same data is displayed in the Nurse Station Application. Hence, the system is highly reliable and accurate.

6.11 Workflow Control

The Nurse Station Application is deployed with CI/CD pipeline (DevOps). Thus, whenever any bug is introduced into the system and the developers commit the patch. The workflow is configured to automatically publish the changes in just a few minutes.

6.12 Error Handling and Recovery

While committing new changes, the application build is tested for compile time errors and is not published in case of any errors. Run-time errors are also handled through promises and try-catch techniques. In case any error is introduced and the system crashes, as soon as the developers release a fix, the CI/CD pipeline publishes the changes automatically throughout the system.

6.13 Logging and Tracing

The system is deployed under Azure cloud platform and each app service is enabled with unique log stream functionality in Azure cloud. So, at any point of time if there are errors in the system or logging and tracing is needed, it can easily be done from the log stream section in Azure cloud.

6.14 Parallelization and Threading

The Nurse Station Web Application uses asynchronous functions for API calls and rendering UI which are both executed in separate threads. Similarly, the Nurse Station Web Services also use asynchronous functions along with a non-blocking input/output structure to serve request parallelly.

6.15 Internationalization

Currently, the Nurse Station Application is capable of showing figures in metric units only.

6.16 Localization

The system only supports English language.

6.17 Communication between Distributed Components

The communication between all the components i.e., IoT, Tablet, Cloud, Web and Mobile Application happens in real-time through SignalR. The IoT device passes the telemetry data to the Tablet which calls an Azure function which saves the data into the database. At the same time, a message is passed via SignalR to notify the Web and Mobile application of this change in data so that they can reflect the same to the user.

6.18 Migration

The database has the ability to handle migration. As the system is cloud based, we can easily upgrade or downgrade the version of database from the Azure panel. Also, both technologies, ReactJs and NodeJs, can accept the migration very easily and the system has the ability to manage migration on the frontend and backend.

7. DESIGN DECISIONS

As, this architecture majorly has 6 or 7 different parts. Let us understand this from the diagram.

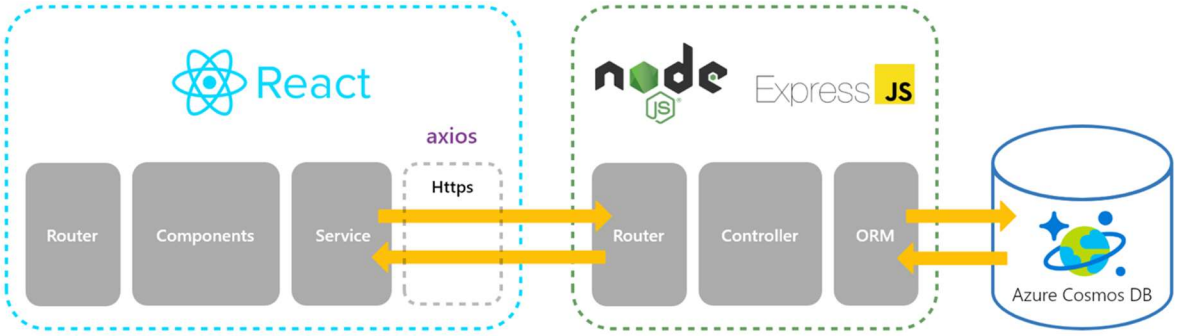

Figure 11: Architecture Design

This is the ReactJs architecture at the frontend which majorly comprises of three parts. The first part has the router which actually helps the users to navigate to different pages. The second part is the components which collectively make up the User Interface of the application. The components help in reducing the redundancy in code and can be swapped easily with new components while upgrading. The service portion of the application is used to make API calls to the backend service.

The system makes the API calls with the help of axios library which helps to fetch the data from the database with the help of another layer which is our backend.

The backend (shown in the middle) communicates with the database with the use of NodeJs and ExpressJs. It will communicate with the database and get the data and send it to the frontend using REST APIs.

CosmosDB, a service of Azure, is a No-SQL database, similar to MongoDB, where the data is stored in collections instead of tables. The data stored in collections is in JSON format which makes it easy to retrieve and understand. Unlike RDBMS, there is no relation between collections and the data in them, thus eliminating foreign key maintenance.

This entire architecture is setup in Azure cloud so, the entire architecture is scalable enough to make it more promising in the future as and when required.

8. LIST OF SOFTWARE OF UNKNOWN PROVENANCE (SOUP)

A separate document listing all the SOUP items is created with document number D001020106 and has all the information about libraries used in the Nurse Station Web Application and Nurse Station Web Services.

9. DEVELOPMENT ENVIRONMENT

9.1 Standards

Nurse Station design and development will follow the SGTC SDLC procedure SGTC-QPR-DLC-001.

9.2 Methods

Not Applicable as it is optional for Class A Device.

9.3 Object-Oriented Design and Analysis with UML

- Client-side
  - ReactJs – Javascript library for building UI
  - Redux – Library for application state management
  - CSS – Styling
- Server-side
  - NodeJs – Javascript runtime
  - ExpressJs – Framework for developing REST API
- Database
  - Cosmos DB – NoSQL database

9.4 Tools

- Visual Studio Code (Open source) for code development
- Azure Cloud.

9.5 Supporting Items

N/A

9.6 Document Revision History

Revision Level	Revision Date	Effective Date	Reason and Description of Revision
00	30th Aug 2021	30th Aug 2021	Initial Release DR1-4 Document was reviewed but not approved and archived, thus archiving

10. APPENDIX A

10.1 Device to Tablet Protocol

List of Events

1. Device register event (Device connection with tablet)
2. Device config event (Device configuration stored in DB)
3. Telemetry Event
4. Heartbeat event (Handles response from tablet and checks connection between tablet and device)
5. Admit Command (Device connected/Device Online)
6. Discharge Command
7. Get weight Command
8. Error Command
9. Tare Command
10. Position Command
11. Position Timer Command
12. Usage Command
13. Firmware-log Command

1. Device register event

```
{
  "event": "Register-Device",
  "deviceId ": "2937987274",
  "unixDateTime": 12323462678,
}
```

2. Device config event

```
{
  "event": "Config-Device",
  "unixDateTime": 12323462678,
  "deviceId": "123-234-562678",
  "positionTimer": 2,
  "telemetryInterval": 5,
  "heartbeatInterval": 4,
  "subCPU1": "1.0",
  "subCPU2": "1.0",
  "subCPU3": "1.0",
  "subCPU4": "1.0",
  "display": "1.0"
}
```

3. Telemetry Event

```
{
  "event": "Telemetary",
  "deviceId": "123-234-562678",
  "tare": 8.17,
  "unixDateTime": 12323462678,
   "wp1": {
     "bedAngle": 1,
     "wc1": 5,
     "wc2": 5,
```

```
      "wc3": 4,
      "wc4": 8
    },
    "wp2": {
      "bedAngle": 2,
      "wc1": 1,
      "wc2": 10,
      "wc3": 5,
      "wc4": 5,
    },
    "wp3": {
       "bedAngle": 3,
       "wc1": 2,
       "wc2": 5,
       "wc3": 5,
       "wc4": 5
    },
    "wp4": {
       "bedAngle": 4,
       "wc1": 5,
       "wc2": 10,
       "wc3": 0,
       "wc4": 5
    }
}
```


4. Heartbeat event

Request (Device to Tablet):


```
{
   "event": "Heartbeat-Event",
   "deviceId": "123-234-562678",
   "unixDateTime": 12323462678
}
```


Response:


```
{
   "Response": [
      {
         "event": "REGISTERRESPOSE",
     "unixDateTime": 12323462678,
      },
      {
         "event": "POSITIONCHANGE",
     "unixDateTime": 12323462678,
         "data": {
            "position": "",
            "positionStatus": 1
         }
      },
      {
         "event": "LOCATEME",
     "unixDateTime": 12323462678,
```

```
        "data": {
          "locateMe": true
        }
      },
      {
        "event": "LOGFILES",
    "unixDateTime": 12323462678,
        "data": {
          "logFiles": true
        }
      }
  ]
}
```


5. Admit Command

Device to Tablet Command


```
{
  "event": "Admit-Message",
  "deviceId": "123-234-562678",
  "unixDateTime": 12323462678,
}
```


6. Discharge Command

Device to Tablet Command


```
{
  "event": "Discharge-Message",
  "deviceId": "123-234-562678",
  "unixDateTime": 12323462678
}
```


7. Get weight Command

Device to Tablet Command


```
{
  "event": "Weight-Message",
  "weight": 50,
  "deviceId": "123-234-562678",
  "unixDateTime": 12323462678
}
```


8. Error Command

Device to Tablet Command


```
{
  "event": "Error-Message",
  "deviceId": "123-234-562678",
  "errordescription": "",
  "bedStatus": 2,
}
```

9. Tare Command

Device to Tablet Command


```
{
   "event": "Tare-Message",
   "deviceId": "123-234-562678",
   "tare": 8.7,
   "unixDateTime": 12323462678
}
```


10. Position Command

Device to Tablet Command


```
{
   "event": "Position-Message",
   "deviceId": "123-234-562678",
   "position_timer": "2",
   "unixDateTime": 12323462678
}
```


11. Position Timer Command

Device to Tablet Command


```
{
   "event": "Position-Timer",
   "deviceId": "123-234-562678",
   "position_timer": "2",
   "unixDateTime": 12323462678
}
```


12. Usage Command

Device to Tablet Command.


```
{
   "event": "Usage-Message",
   "deviceId": "123-234-562678",
   "eventType": "GetWeight",
   "unixDateTime": 12323462678
}
```


13. Firmware-log Command

Device to Tablet Command


```
{
   "event": "Log-Message",
   "deviceId": "123-234-562678",
   "unixDateTime": 12323462678
}
```


10.2 **Tablet to IoT Protocol**

1. Gateway Device register event

Tablet Data


```
{
   "event": "Register-GatewayDevice",
   "tabletId": "tablet-1",
   "hospital_id": "eb456c01-fc4b-4112-b674"
}
```


Final JSON


```
{
   "tablet_id": "tablet-1",
   "hospital_id": "eb456c01-fc4b-4112-b674",
   "device_unix_time": 12323462678
}
```


Set Twin Property


```
{
 "reported": {
   "VersionInfo": {
     "BuildNo": "1.0.0.0.11",
     "OSVersion": "11",
     "SettingsAppVersion":"1.0",
     "ManagementAppVersion":"1.0",
     "KioskAppVersion":"1.0",
     "ICUAppVersion":"1.0",
     "SoftwareUpdateAppVersion":"1.0",
   }
  }
}
```


2. Gateway Device connection status


```
{
   "event": "GatewayDevice-ConnectionStatus",
   "tablet_id": "tablet-1",
   "connected": 1
}
```


3. Device register event

Device Data


```
{
 "event": "Register-Device",
 "deviceId ": "2937987274",
 "unixDateTime": 12323462678,
}
```


Tablet Data


```
{
   "tabletId": "tablet-1",
   "hospital_id": "eb456c01-fc4b-4112-b674"
```

}

Final JSON
{
"tablet_id": "tablet-1",
"device_id": "2937987274",
"hospital_id": "eb456c01-fc4b-4112-b674",
"device_unix_time": 12323462678
}

4. Device connection status

{
"event": "Device-ConnectionStatus",
"device_id": "2937987274",
"connected": 1
}

5. Device config event

{
"event": "Config-Device",
"unixDateTime": 12323462678,
"telemetryInterval": 5,
"heartbeatInterval": 4,
"positionTimer": 2,
"deviceId": "123-234-562678",
"subCPU1": "1.0",
"subCPU2": "1.0",
"subCPU3": "1.0",
"subCPU4": "1.0",
"display": "1.0"
}

Note: Set above device twin property in DB

6. Factory reset event

{
"event": "Usage-Message",
"serial_number": "123-234-562678",
"event_type": "FactoryReset",
"date_time": "123-234-562678"
}

7. Telemetry Event

Device Data

{
"event": "Telemetary",
"deviceId": "123-234-562678",
"tare": 8.17,
"unixDateTime": 12323462678,
"wp1": {
"bedAngle": 1,

```
      "wc1": 5,
      "wc2": 5,
      "wc3": 4,
      "wc4": 8
    },
    "wp2": {
      "bedAngle": 2,
      "wc1": 1,
      "wc2": 10,
      "wc3": 5,
      "wc4": 5,
    },
    "wp3": {
      "bedAngle": 3,
      "wc1": 2,
      "wc2": 5,
      "wc3": 5,
      "wc4": 5
    },
    "wp4": {
      "bedAngle": 4,
      "wc1": 5,
      "wc2": 10,
      "wc3": 0,
      "wc4": 5
    }
}
```


Tablet Data

Note: All values here except date and episodeId are static


```
{
  "patient_position": 3,
  "datetime": "Thu Jul 9 09:54:44 2020 GMT",
  "episodeID": "episode_ID",
  "telemetaryversion": "1.0",
  "devicetype": "smartmedic",
  "deviceversion": "1.0",
  "isValidSensorDataPresent": true,
  "exitAlarm": "",
  "weight_on_demand": false
}
```


Final JSON

Note: Sensor values will also be included


```
{
  "bedID": "deviceid",
  "patient_position": 3,
  "datetime": "DateTime.Now.ToString()",
  "episodeID": "episode_ID",
  "soft_tare": 8.77,
  "telemetaryversion": "1.0",
  "devicetype": "smartmedic",
```

"deviceversion": "1.0",
"isValidSensorDataPresent": true,
"exitAlarm": "",
"weight_on_demand": false
}

8. Admit Command

Device Data

{
"event": "Admit-Message",
"deviceId": "123-234-562678",
"unixDateTime": 12323462678,
}

Tablet Data

{
"hospitalCode": "56789",
"episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
"bed_status": 1,
"icu_id": "79cb30aa-817c-4ec1-823c-9d4c6f2ccc86",
"bed_Name": "004",
"position_timer": "2"
}

Final JSON

{
"eventName": "Admit-Message",
"hospitalCode": "56789",
"tabletId": "tablet-1",
"bedID": "123-234-562678",
"episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
"bed_status": 1,
"icu_id": "79cb30aa-817c-4ec1-823c-9d4c6f2ccc86",
"bed_Name": "004",
"position_timer": "2"
}

9. Discharge Command

Device Data

{
"event": "Discharge-Message",
"deviceId": "123-234-562678",
"unixDateTime": 12323462678
}

Tablet Data

{
"episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
"bed_status": 0
}

Final JSON


```
{
  "eventName": "Discharge-Message",
  "messageversion":"2.0",
  "bedID": "123-234-562678",
  "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
  "bed_status": 0
}
```


10. Position Command

Device Data


```
{
  "event": "Position-Message",
  "deviceId": "123-234-562678",
  "position_timer": "2",
  "unixDateTime": 12323462678
}
```


Tablet Data


```
{
  "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
  "position": "Supine",// "Supine","Left Lateral","Right Lateral"
  "position_status": 2, // 0,1,2
  "date_time": "10/20/2021 9:36:28 AM"
}
```


Final JSON


```
{
  "eventName": "Position-Message",
  "bedID": "123-234-562678",
  "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
  "position": "Supine",// "Supine","Left Lateral","Right Lateral"
  "position_status": 2, // 0,1,2
  "position_timer": "2",
  "date_time": "10/20/2021 9:36:28 AM"
}
```


11. Get weight Command

Device Data


```
{
  "event": "Weight-Message",
  "weight": 50,
  "deviceId": "123-234-562678",
  "unixDateTime": 12323462678
}
```


Tablet Data


```
{
  "hospitalCode": "56789",
```

```
  "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
  "devicetype": "smartmedic",
  "deviceversion": "1.0",
  "messageversion": "1.0",
  "sequencenumber": "1"
}
```


Final JSON

Note: Sensor values will also be included


```
{
  "eventName": "Weight-Message",
  "hospitalCode": "56789",
  "bedID": "123-234-562678",
  "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
  "weight": 60,
  "unixdatetime": 1611392325,
  "devicetype": "smartmedic",
  "deviceversion": "1.0",
  "messageversion": "1.0",
  "sequencenumber": "1"
}
```


12. Error Command

Device Data


```
{
  "event": "Error-Message",
  "deviceId": "123-234-562678",
  "errordescription": "",
  "bedStatus": 2,
}
```


Final JSON


```
{
  "eventName": "Error-Message",
  "bedID": "123-234-562678",
  "errordescription": "",
  "bed_status": 2,
}
```


13. Tare Command

Device Data


```
{
  "event": "Tare-Message",
  "deviceId": "123-234-562678",
  "tare": 8.7,
  "unixDateTime": 12323462678
}
```

Tablet Data
Note: All values here except bedName, hospitalCode, icuId, and episodeID are static


```
{
   "bed_name": "004",
   "hospitalcode": "56789",
   "icu_id": "79cb30aa-817c-4ec1-823c-9d4c6f2ccc86",
   "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
}
```


Final JSON
Note: Sensor values will also be included


```
{
   "eventName": "Tare-Message",
   "messageversion":"2.0",
   "bedID": "123-234-562678",
   "bed_name": "004",
   "hospitalcode": "56789",
   "icu_id": "79cb30aa-817c-4ec1-823c-9d4c6f2ccc86",
   "unixdatetime": 1611392325,
   "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
   "Tarevalue": 8.7
}
```


14. Firmware-log Command

Device Data


```
{
   "event": "Log-Message",
   "deviceId": "123-234-562678",
   "unixDateTime": 12323462678
}
```


15. Position Timer Command

Device Data


```
{
   "event": "Position-Timer",
   "deviceId": "123-234-562678",
   "position_timer": "2",
   "unixDateTime": 12323462678
}
```


Tablet Data


```
{
   "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739"
}
```


Final JSON


```
{
```

```
  "eventName": "Position-Timer",
  "bedID": "123-234-562678",
  "episodeID": "eb456c01-fc4b-4112-b674-5bab06993739",
  "position_timer": "2"
}
```


11. APPENDIX B

admin_users: partitionKey(partition Key)


```
{
  "name": "Stryker Admin",
  "first_name": "Stryker",
  "last_name": "Admin",
  "email_id": "test@stryker.com",
  "password": "U3RWtlckAyMDIx",
  "is_active": 1,
  "is_notify_report": 1,
  "is_deleted": 0,
  "attempts_done": 0,
  "last_attempt_time": 0,
  "allattempts_withintime": 0,
  "role": 1,
  "hospital_id": "",
  "contact_number": "+91",
  "created_time": 1615463555,
  "email_token": "I5wbj3t7aQ7l2sLOPy7PpOQeSq3q2yI1nwIHTAUKEU4vDMjYKbnDH8=",
  "id": "44282414-f207-40d1-9970-cf7c64a0dc57",
  "_rid": "CTdpAPiv5mMBAAAAAAAAAA==",
  "_self": "dbs/CTdpAA==/colls/CTdpAPiv5mM=/docs/CTdpAPiv5mMBAAAAAAAAAA==/",
  "_etag": "\"f701959f-0000-1800-0000-61cae5b70000\"",
  "_attachments": "attachments/",
  "_ts": 1640687031
}
```


device_config: serial_number(partition Key)


```
{
  "serial_number": "Thus-1",
  "subCPU1": "1.0",
  "subCPU2": "1.0",
  "subCPU3": "1.0",
  "subCPU4": "1.0",
  "display": "1.1",
  "id": "84c5060d-51ec-44ef-b47e-e6c6ca7e813c",
  "date_time": "10/7/2021 2:41:39 PM",
  "unix_date_time": 1633617699,
  "_rid": "CTdpAJ1NhZkBAAAAAAAAAA==",
  "_self": "dbs/CTdpAA==/colls/CTdpAJ1NhZk=/docs/CTdpAJ1NhZkBAAAAAAAAAA==/",
  "_etag": "\"08009827-0000-1800-0000-61bae76d0000\"",
  "_attachments": "attachments/",
  "_ts": 1639638893
}
```

device_software_upgrades: partitionKey(partition Key)


```
{
   "file_type": 1,
   "application_version": "1.0.2",
   "description": "Software Upgrade 1.0.2",
   "file_name": "software_v1.0.2.zip",
   "is_active": 0,
   "created_time": "Mon, 15 Nov 2021 11:40:36 GMT",
   "created_unix_time": 1636976436,
   "id": "a4c73ce0-2f92-4060-b4ca-fcfd9d45b3cd",
   "_rid": "CTdpAI4Lql0BAAAAAAAAAA==",
   "_self": "dbs/CTdpAA==/colls/CTdpAI4Lql0=/docs/CTdpAI4Lql0BAAAAAAAAAA==/",
   "_etag": "\"0f00e55d-0000-1800-0000-6194d4cb0000\"",
   "_attachments": "attachments/",
   "_ts": 1637143755
}
```


user_audit_log: partitionKey(partition Key)


```
{
   "LoginStatusResponse": {
      "DateTime": "2021-12-16 09:23:45",
      "Location": {
         "Country": "",
         "State": "",
         "Postal": ""
      },
      "NetworkType": "Reliance",
      "ApiResponse": {
         "StatusCode": 200,
         "URL": "https://demo-nurseStation",
         "Method": "post"
      },
      "Path": "/",
      "IpAddress": "10.0.3.119",
      "SessionTime": "",
      "LoginStatus": false
   },
   "created_time": 1639643877,
   "id": "1c27de6a-7877-4bdb-b067-227c7095b09d",
   "_rid": "CTdpAJsOYNECAAAAAAAAAA==",
   "_self": "dbs/CTdpAA==/colls/CTdpAJsOYNE=/docs/CTdpAJsOYNECAAAAAAAAAA==/",
   "_etag": "\"1700a267-0000-1800-0000-61bafae60000\"",
   "_attachments": "attachments/",
   "_ts": 1639643878
}
```


gateway_devices: tablet_id (partition Key)


```
{
   "tablet_id": "--tablet73f8eede0c4763bc",
   "hospital_id": "1f90a31d-b478-4fac-a683-c2652c45e653",
   "icu_id": "467949b7-6529-47ac-8dcf-c89772ae4b1e",
   "connection_status": 1,
```

```
  "created_at_unix": 1640675481,
  "created_at": "12/28/2021 7:11:21 AM",
  "id": "0f1d0285-9f70-4fc6-afee-441783a77410",
  "date_time": "12/24/2021 10:04:14 AM",
  "unix_date_time": 1640340254,
  "_rid": "CTdpAN32EE5HAAAAAAAAAA==",
  "_self": "dbs/CTdpAA==/colls/CTdpAN32EE4=/docs/CTdpAN32EE5HAAAAAAAAAA==/",
  "_etag": "\"13043047-0000-1800-0000-61cb129b0000\"",
  "_attachments": "attachments/",
  "_ts": 1640698523
}
```


bed_master : icu_id (partition key)


```
{
  "bed_name": "003",
  "bed_desc": "",
  "icu_id": "6e1535bc-94c7-4218-9ab3-35834d3b162b",
  "is_active": 1,
  "created_time": 1590491473,
  "id": "266d2a8d-5bf5-4c99-9453-97c312f413f5",
  "_rid": "A10bAJDEoWKBhB4AAAAAAA==",
  "_self": "dbs/A10bAA==/colls/A10bAJDEoWI=/docs/A10bAJDEoWKBhB4AAAAAAA==/",
  "_etag": "\"00003401-0000-0700-0000-5ed7d71d0000\"",
  "_attachments": "attachments/",
  "_ts": 1591203613
}
```


configuration : hospital_id (partition key)


```
{
  "cooling_period_minute": 3,
  "no_of_beds": "16",
  "max_position_timer": 8,
  "update_timer_minute": 3,
  "cron_serial_numbers": ["sn-007-888-abc-mac-a1-b2-c3-d4-e5-n1"],
  "errorNotificationEmail": ["harsh.machhoya@thegatewaytest.com"],
  "cron_emails": ["harsh.machhoya@thegatewaytest.com"],
  "cron_no_of_days": 3,
  "cron_no_of_clicks": 5,
  "cron_disconnected_last_days": 1,
  "fr_sos_time_limit_seconds": 300,
  "telemetary_eventtypes": [
    "Tare-Message",
    "OverrideTare-Message",
    "PatientStatus-Message",
    "start-Message"
  ],
  "email_configure": {
    "email_is_active": 1,
    "report_list": [
      {
        "id": 1,
        "name": "Weight Data"
```

```
    },
    {
      "id": 2,
      "name": "Weight Data: Red Flag"
    }
  ],
  "schedules": [
    {
      "id": 1,
      "type": "Daily",
      "report": [
        {
          "report_id": 1,
          "is_daily_active": 1,
          "unix_time": 52200,
          "users_ids": [
            "*",
            "512f04b5-20fd-4c4b-9f42-31387e91c923",
          ]
        },
        {
          "report_id": 2,
          "is_daily_active": 1,
          "unix_time": 52200,
          "users_ids": [
            "ba075cc4-1a9c-4e93-a82e-50074c5a4b4b"
          ]
        }
      ]
    },
    {
      "id": 2,
      "type": "Weekly",
      "report": [
        {
          "report_id": 1,
          "is_weekly_active": 1,
          "week_day": 1,
          "unix_time": 54000,
          "users_ids": [
            "ba075cc4-1a9c-4e93-a82e-50074c5a4b4b"
          ]
        },
        {
          "report_id": 2,
          "is_weekly_active": 1,
          "week_day": 1,
          "unix_time": 56100,
          "users_ids": [
            "ba075cc4-1a9c-4e93-a82e-50074c5a4b4b"
          ]
        }
```

```
            ]
          }
        ]
    },
    "jwt_private_key": "privatekey",
    "jwt_token_expires_sec": 600,
    "id": "ad6ca6d6-2d8b-44fb-b459-cada976c50fb",
    "_rid": "CTdpALwW8H0BAAAAAAAAAA==",
    "_self": "dbs/CTdpAA==/colls/CTdpALwW8H0=/docs/CTdpALwW8H0BAAAAAAAAAA==/",
    "_etag": "\"f701b5e7-0000-1800-0000-61c9aa880000\"",
    "_attachments": "attachments/",
    "maximum_login_attempts": 3,
    "next_login_attempt_minute": 15,
    "within_attempts_minute": 2,
    "_ts": 1640606344
}
```


device-telemetary: bedID (partition key)


```
{
    "dynObj": null,
    "SequenceNumber": "2",
    "telemetaryversion": "1.0",
    "messageversion": null,
    "devicetype": "smartmedic",
    "deviceversion": "1.0",
    "hospitalcode": null,
    "episodeID": "4053fa76-8fb5-437b-b44e-1a19c8f43665",
    "bedID": "a9ka5uao2nq5iybj20zl",
    "messagetype": null,
    "isValidSensorDataPresent": "True",
    "patient_position": "3",
    "exitAlarm": "",
    "weight_on_demand": "False",
    "datetime": "Tue Dec 28 19:32:24 2021",
    "unixdatetime": 1640700144,
    "id": "628b2426-cb58-48cc-a4d4-19b8bb9447ce",
    "soft_tare": -0.0250653624534607,
    "wp1": {
        "BedAngle": 0.247,
        "WC1": 0.175,
        "WC2": 1.761,
        "WC3": 2.518,
        "WC4": 0.659
    },
    "wp2": {
        "BedAngle": 0,
        "WC1": 0,
        "WC2": 0,
        "WC3": 0,
        "WC4": 0
    },
    "wp3": {
```

```
      "BedAngle": 0,
      "WC1": 0,
      "WC2": 0,
      "WC3": 0,
      "WC4": 0
   },
   "wp4": {
      "BedAngle": 0,
      "WC1": 0,
      "WC2": 0,
      "WC3": 0,
      "WC4": 0
   },
   "_rid": "CTdpAIe06bFcE6wAAAAAAA==",
   "_self": "dbs/CTdpAA==/colls/CTdpAIe06bE=/docs/CTdpAIe06bFcE6wAAAAAAA==/",
   "_etag": "\"4900c198-0000-1800-0000-61cb18f20000\"",
   "_attachments": "attachments/",
   "_ts": 1640700146
}
```


devices : serial_number (partition key)


```
{
   "bed_name": "001",
   "bed_desc": "",
   "serial_number": "Device1",
   "icu_id": "c7a7a988-9def-4c46-93f8-40305354d73f",
   "tablet_id": null,
   "bed_status": 2,
   "position_timer": "2",
   "telemetryInterval": 0,
   "connection_status": 0,
   "id": "43e66567-0020-4d2f-9b7d-b25447bf0177",
   "date_time": "9/14/2021 11:30:13 AM",
   "unix_date_time": 1631619013,
   "_rid": "CTdpALy3AuILAAAAAAAAAA==",
   "_self": "dbs/CTdpAA==/colls/CTdpALy3AuI=/docs/CTdpALy3AuILAAAAAAAAAA==/",
   "_etag": "\"9d00c87d-0000-1800-0000-61a62e690000\"",
   "_attachments": "attachments/",
   "_ts": 1638280809
}
```


episode_details : bedID (partition key)


```
{
   "bedID": "101 mobile",
   "is_active": 1,
   "reference_number": null,
   "id": "21f0e8b6-fe2d-4bba-8986-e0d3d872eb48",
   "date_time": "12/28/2021 1:53:35 PM",
   "unix_date_time": 1640699615,
   "_rid": "CTdpAKacJY-tBAAAAAAAAA==",
   "_self": "dbs/CTdpAA==/colls/CTdpAKacJY8=/docs/CTdpAKacJY-tBAAAAAAAAA==/",
   "_etag": "\"4000160d-0000-1800-0000-61cb16df0000\"",
```

```
  "_attachments": "attachments/",
  "_ts": 1640699615
}
```


error_messages : serial_number (partition key)


```
{
  "serial_number": "123-234-1988-009",
  "error_desc": "tareWeight is failed.",
  "id": "cf9d63b2-21eb-432c-ae9c-df1e0c11cfae",
  "date_time": "7/9/2020 9:53:44 AM",
  "unix_date_time": 1594288424,
  "_rid": "A10bAMR3wiyChB4AAAAAAA==",
  "_self": "dbs/A10bAA==/colls/A10bAMR3wiw=/docs/A10bAMR3wiyChB4AAAAAAA==/",
  "_etag": "\"1b01377e-0000-0700-0000-5f06e92b0000\"",
  "_attachments": "attachments/",
  "_ts": 1594288427
}
```


hospital_master : hospital_master (partition key)


```
{
  "hospital_name": "Apollo",
  "hospital_desc": "",
  "hospital_code": "123456",
  "email": "",
  "hospital_address": {
    "Address": "",
    "City": "Surat",
    "State": "",
    "PinCode": "",
    "Country": "India",
    "Area": "Pakwan"
  },
  "contact_number": "",
  "admin_code": "123456",
  "api_key": "v3GFko/iTfCxTpbXLVTstnpKuqXHrEuuCBvAO0oc4wMLWUp6nFKyT2Htwfw8OpN+",
  "valid_till": "",
  "is_active": 1,
  "email_recipients": [],
  "created_time": "Tue, 14 Sep 2021 10:47:29 GMT",
  "created_unix_time": 1631616449,
  "id": "712fa54a-aa1b-4e7c-b3ba-a10642c908eb",
  "_rid": "CTdpAKxmalYCAAAAAAAAAA==",
  "_self": "dbs/CTdpAA==/colls/CTdpAKxmalY=/docs/CTdpAKxmalYCAAAAAAAAAA==/",
  "_etag": "\"8a01b282-0000-1800-0000-615e7c9d0000\"",
  "_attachments": "attachments/",
  "_ts": 1633582237
}
```


icu_master : hospital_id (partition key)


```
{
  "hospital_id": "f9ed1c5d-f981-4076-a9ef-0ae4346c8e76",
  "icu_name": "Cardiac ICU",
```

"no_of_selected_beds": 2,
"is_active": 1,
"created_time": 1590675809,
"id": "194c19fb-8b99-4863-868f-0076ccf31d3c",
"updated_time": 1590750578,
"_rid": "A10bAJOhuJ+BhB4AAAAAAA==",
"_self": "dbs/A10bAA==/colls/A10bAJOhuJ8=/docs/A10bAJOhuJ+BhB4AAAAAAA==/",
"_etag": "\"0c00ff3e-0000-0700-0000-5ed7d7740000\"",
"_attachments": "attachments/",
"_ts": 1591203700
}

turn_data : bedID (partition key)

{
"bedID": "sn-007-888-abc-mac-a1-b2-c3-d4-e5-d7",
"episode_id": "7e178746-7858-4106-bfa6-85a5b52b75ff",
"position": "Right Lateral",
"position_status": 2,
"position_timer": "2",
"connection_off": 0,
"id": "a6f27c94-8498-41a4-93c6-a9c93d100c51",
"date_time": "5/26/2020 2:17:33 PM",
"unix_date_time": 1590422400,
"_rid": "A10bAIEBetmBhB4AAAAAAA==",
"_self": "dbs/A10bAA==/colls/A10bAIEBetk=/docs/A10bAIEBetmBhB4AAAAAAA==/",
"_etag": "\"2e00f235-0000-0700-0000-5ed7d8510000\"",
"_attachments": "attachments/",
"_ts": 1591203921
}

usuage_data : serial_number (partition key)

{
"serial_number": "860536043525924",
"event_type": "GetWeight",
"id": "eabebf1c-b03d-4a6e-95c3-550a96f50c73",
"date_time": "7/27/2020 10:19:06 AM",
"unix_date_time": 1595845146,
"_rid": "A10bAImGSz+qih4AAAAAAA==",
"_self": "dbs/A10bAA==/colls/A10bAImGSz8=/docs/A10bAImGSz+qih4AAAAAAA==/",
"_etag": "\"7c010668-0000-0700-0000-5f1eaa1b0000\"",
"_attachments": "attachments/",
"_ts": 1595845147
}

weight_data : bedID (partition key)

{
"bedID": "sn-007-888-abc-mac-a1-b2-c3-d4-e5-d7",
"episode_id": "7e178746-7858-4106-bfa6-85a5b52b75ff",

```
  "weight_data": 75.5,
  "connection_off": 0,
  "id": "5ac091fe-6457-41be-bcb2-648846eabf55",
  "date_time": "5/26/2020 2:21:58 PM",
  "unix_date_time": 1590471000,
  "_rid": "A10bAJ54W8eChB4AAAAAAA==",
  "_self": "dbs/A10bAA==/colls/A10bAJ54W8c=/docs/A10bAJ54W8eChB4AAAAAAA==/",
  "_etag": "\"4b00cb2d-0000-0700-0000-5ed7d82d0000\"",
  "_attachments": "attachments/",
  "_ts": 1591203885
}
```

12. **APPENDIX C**

Software Architecture Design Checklist document as per SGTC template SGTC-QFM-DLC-001-05 Ver 07 reviewed and approved in this document**.**

Area of Concern	Item to be Verified	Is Implemented	Not Applicable	Comments
Software Requirement	Is the Software Requirements Documentation completed? (Note: Completion implies it contains adequate content that it allows the process to move forward).	☑	☐	
Architecture Design	Is the overall organization of the program clear, including a good architectural overview and justification?	☑	☐	
Architecture Design	Is the architecture design complete? Are all the Software Requirements covered? All Risk Control related software requirements are covered?	☑	☐	
Architecture Design	Are all critical classes described and justified? (Relevant in case of Object Oriented Design).	☐	☑	SmartMedic is Class A: NA
Architecture Design	Is the data design defined and justified?	☑	☐	
Architecture Design	Is the database organization and content specified?	☑	☐	
Architecture Design	Is the strategy for UI design described?	☑	☐	
Architecture Design	Is the UI modularized so that changes in one do not affect the others?	☑	☐	
Architecture Design	Are the strategies and approaches for following are defined? Memory Management. Internationalization.	☑	☐	
Documentation	Are all the relevant architectural views documented? Logical View (Class diagram or Data Model) Process View (Thread Controlling) Physical View (Diagram depicting all components such as software and hardware) Development View (How the code is to be organized)	☑	☐	
Documentation	Was the document baseline completed before review?	☑	☐	
Components	Are all the architectural components well defined?	☑	☐	

Area of Concern	Item to be Verified	Is Implemented	Not Applicable	Comments
Implementation	Is the design solution realizable? All components can be implemented and integrated together?	☑	☐	
Implementation	Are you as the programmer or the implementer of the system comfortable with the architecture?	☑	☐	
Interfaces	Are all the interfaces between architectural components well defined? Includes the following: Interfaces between software items. Interfaces between software items and components external to software items. Interfaces between software items and hardware devices.	☑	☐	
SOUP	Is the design supporting proper operation of the Software Items identified as SOUP?	☑	☐	
SOUP	Have the known bugs in the release notes of the SOUP been analyzed and if necessary, addressed in the Software Architectural Design	☐	☑	Not applicable for Class A
Security	Does the design understand, identify and accommodate the company's security policy and the various threats to the application?	☑	☐	
Exception Handling	Is a coherent Exception handling strategy defined?	☑	☐	
I/O	Is the strategy for handling I/O described?	☑	☐	
Risk Control	Is the approach to fault tolerance and Risk Control defined?	☑	☐	
Flexibility	Is the architecture flexible and is able to cope with likely changes in the requirements.	☑	☐	
Scalability	Is the architecture design scalable according to the expectations in the requirements?	☑	☐	
Interoperability	Is the top level design independent of the machine/hardware and language used to implement it?	☑	☐	
Assumptions	Are all the assumptions made documented?	☐	☑	There are no assumptions made
Constraints	Are all the constraints documented?	☐	☑	There are no constraints
Dependencies	Are all the dependencies documented?	☑	☐	

D001020031 Software Architect Diagram SmartMedic Phase II Cloud-Nurse Station

Final Audit Report 2022-04-02

Created:	2022-03-30
By:	Vinod T Londhe (vinod.londhe@stryker.com)
Status:	Signed
Transaction ID:	CBJCHBCAABAA-yJ5v6xeBroNcfHl7DrtrVPLKHGU8kkp

"D001020031 Software Architect Diagram SmartMedic Phase II Cloud-Nurse Station" History

Document created by Vinod T Londhe (vinod.londhe@stryker.com)
2022-03-30 - 7:37:33 AM GMT- IP address: 1.6.9.50

Document emailed to Deepak Sharma (deepak.sharma@stryker.com) for signature
2022-03-30 - 7:43:41 AM GMT

Deepak Sharma (deepak.sharma@stryker.com) verified identity with Adobe Sign authentication
2022-03-31 - 4:18:53 AM GMT

Document e-signed by Deepak Sharma (deepak.sharma@stryker.com)
Signature Date: 2022-03-31 - 4:18:53 AM GMT - Time Source: server- IP address: 150.129.238.45

Document emailed to Akhil Gupta (akhil.gupta@stryker.com) for signature
2022-03-31 - 4:18:56 AM GMT

Email viewed by Akhil Gupta (akhil.gupta@stryker.com)
2022-03-31 - 4:26:19 AM GMT- IP address: 47.31.141.9

Akhil Gupta (akhil.gupta@stryker.com) verified identity with Adobe Sign authentication
2022-04-01 - 3:45:02 AM GMT

Document e-signed by Akhil Gupta (akhil.gupta@stryker.com)
Signature Date: 2022-04-01 - 3:45:02 AM GMT - Time Source: server- IP address: 103.170.152.218

Document emailed to Pragya Nidhi (pragya.nidhi@stryker.com) for signature
2022-04-01 - 3:45:04 AM GMT

Email viewed by Pragya Nidhi (pragya.nidhi@stryker.com)
2022-04-01 - 4:22:22 AM GMT- IP address: 183.83.137.6

Pragya Nidhi (pragya.nidhi@stryker.com) verified identity with Adobe Sign authentication
2022-04-01 - 6:00:40 AM GMT

Document e-signed by Pragya Nidhi (pragya.nidhi@stryker.com)
Signature Date: 2022-04-01 - 6:00:40 AM GMT - Time Source: server- IP address: 183.83.137.6

Document emailed to Vikram Puri (vikram.puri1@stryker.com) for signature
2022-04-01 - 6:00:42 AM GMT

Email viewed by Vikram Puri (vikram.puri1@stryker.com)
2022-04-01 - 8:25:10 AM GMT- IP address: 104.47.56.254

Vikram Puri (vikram.puri1@stryker.com) verified identity with Adobe Sign authentication
2022-04-01 - 8:27:57 AM GMT

Document e-signed by Vikram Puri (vikram.puri1@stryker.com)
Signature Date: 2022-04-01 - 8:27:57 AM GMT - Time Source: server- IP address: 122.161.79.211

Document emailed to Sreejith Viswam (sreejith.viswam@stryker.com) for signature
2022-04-01 - 8:27:59 AM GMT

Sreejith Viswam (sreejith.viswam@stryker.com) verified identity with Adobe Sign authentication
2022-04-02 - 4:03:15 AM GMT

Document e-signed by Sreejith Viswam (sreejith.viswam@stryker.com)
Signature Date: 2022-04-02 - 4:03:15 AM GMT - Time Source: server- IP address: 106.196.18.56

Agreement completed.
2022-04-02 - 4:03:15 AM GMT